Fault-Tolerance, Network Storage and Logistical Computing

James S. Plank

Director:
Logistical Computing and Internetworking (LoCI) Laboratory

Department of Computer Science
University of Tennessee

LoCI Lab Personnel and Funding

Directors:

Jim Plank
Micah Beck

Exec Director:

Terry Moore

Students:

Erika Fuentes
Xiang Li
Sharmila Kancherla
Kent Galbraith

Research Staff:

Scott Atchley
Alexander Bassi
Ying Ding
Hunter Hagewood
Jeremy Millar
Stephen Soltesz
Yong Zheng

Funding:

NSF - NGS, Itech, MWIR, etc.
DOE - SciDAC
UTK - Center of Excellence

Major Collaborators

- Jack Dongarra (UT - NetSolve, Linear Algebra)
- Rich Wolski (UCSB - Network Weather Service)
- Fran Berman (UCSD/NPACI - Scheduling)
- Henri Casanova (UCSD/NPACI - Scheduling)

LoCI:
Logistical Computing and Internetworking

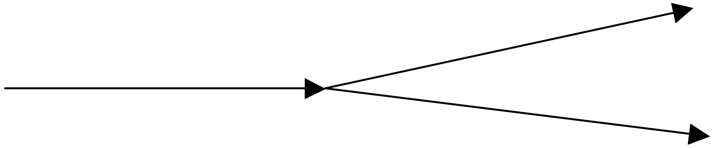
Revolves around the principle of:

Logistical Networking

Allowing applications to manage the *trajectory* of data in space and time as it travels across the network.

Managing Trajectories

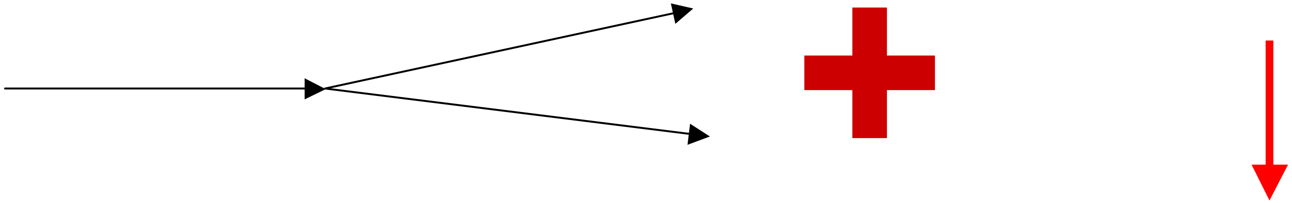
Networking (spatial)



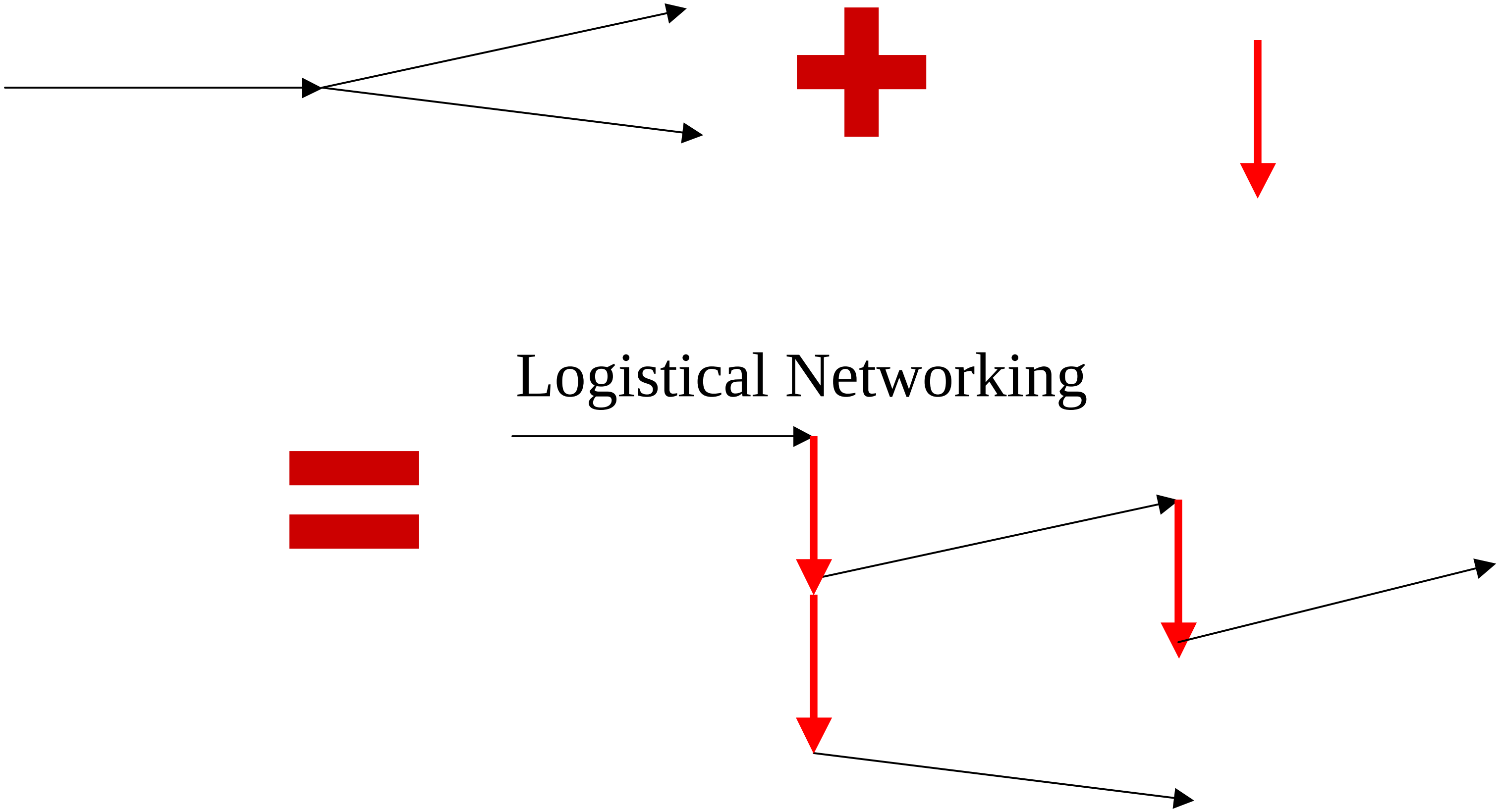
Managing Trajectories

Networking (spatial)

Storage (temporal)



Managing Trajectories



Networking (spatial)

+

Storage (temporal)

=

Logistical Networking

LoCI’s Mission

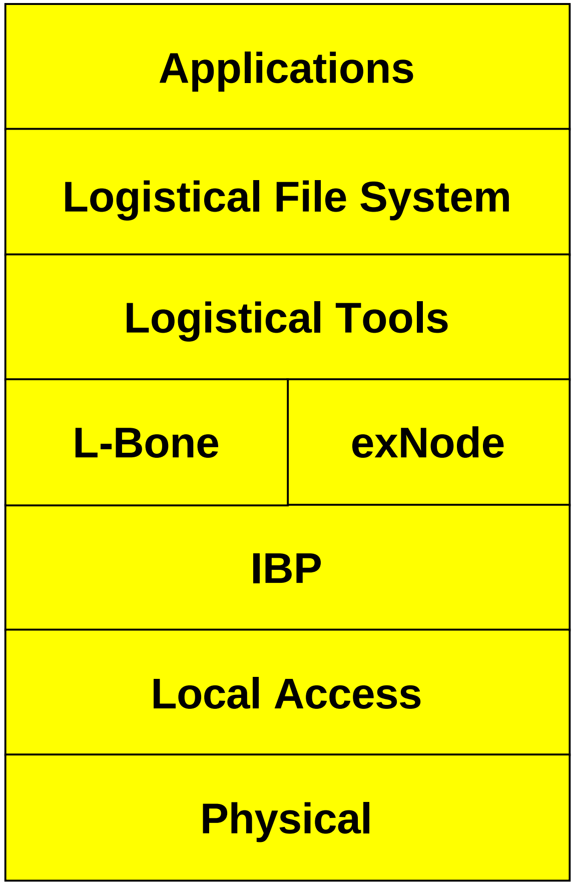
To Improve:

- Application performance
- Application functionality
- Overall resource utilization

As a result of logistical networking.

The Network Storage Stack

- A Fundamental Organizing Principle
- Like the IP Stack
- Each level encapsulates details from the lower levels, while still exposing details to higher levels



Applications

Logistical File System

Logistical Tools

L-Bone

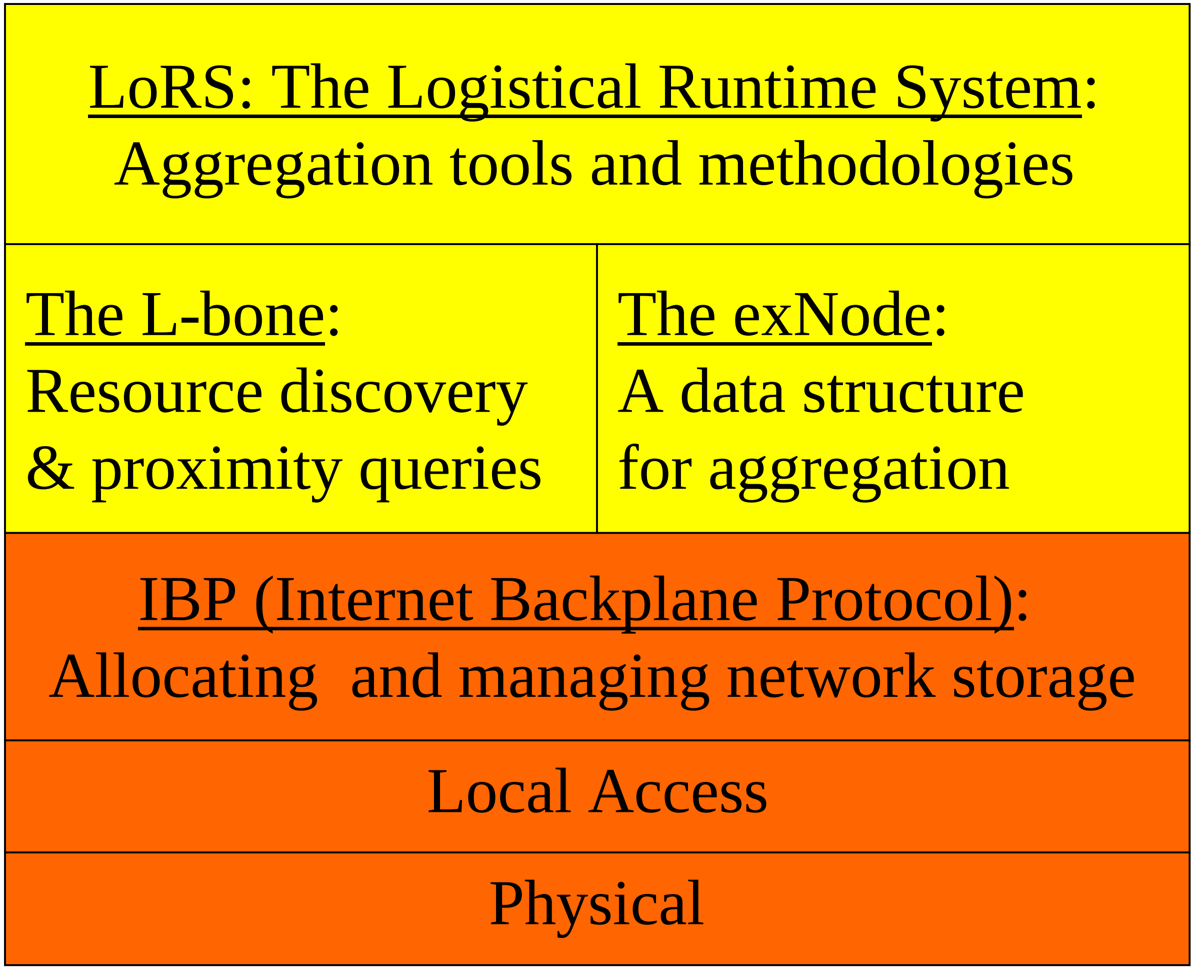
exNode

IBP

Local Access

Physical

The Network Storage Stack



LoRS: The Logistical Runtime System:
Aggregation tools and methodologies

The L-bone:
Resource discovery
& proximity queries

The exNode:
A data structure
for aggregation

IBP (Internet Backplane Protocol):
Allocating and managing network storage

Local Access

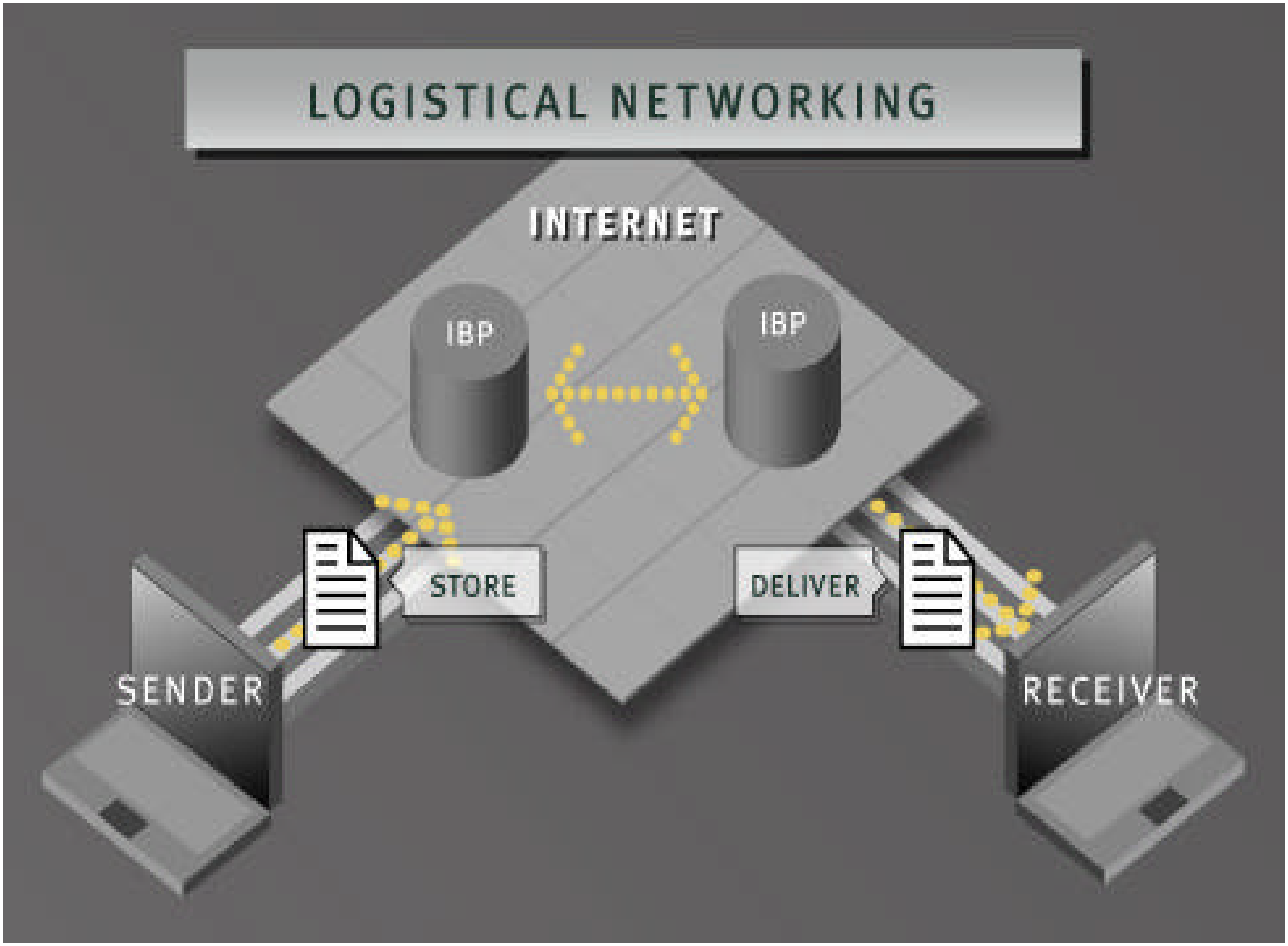
Physical

IBP: The Internet Backplane Protocol

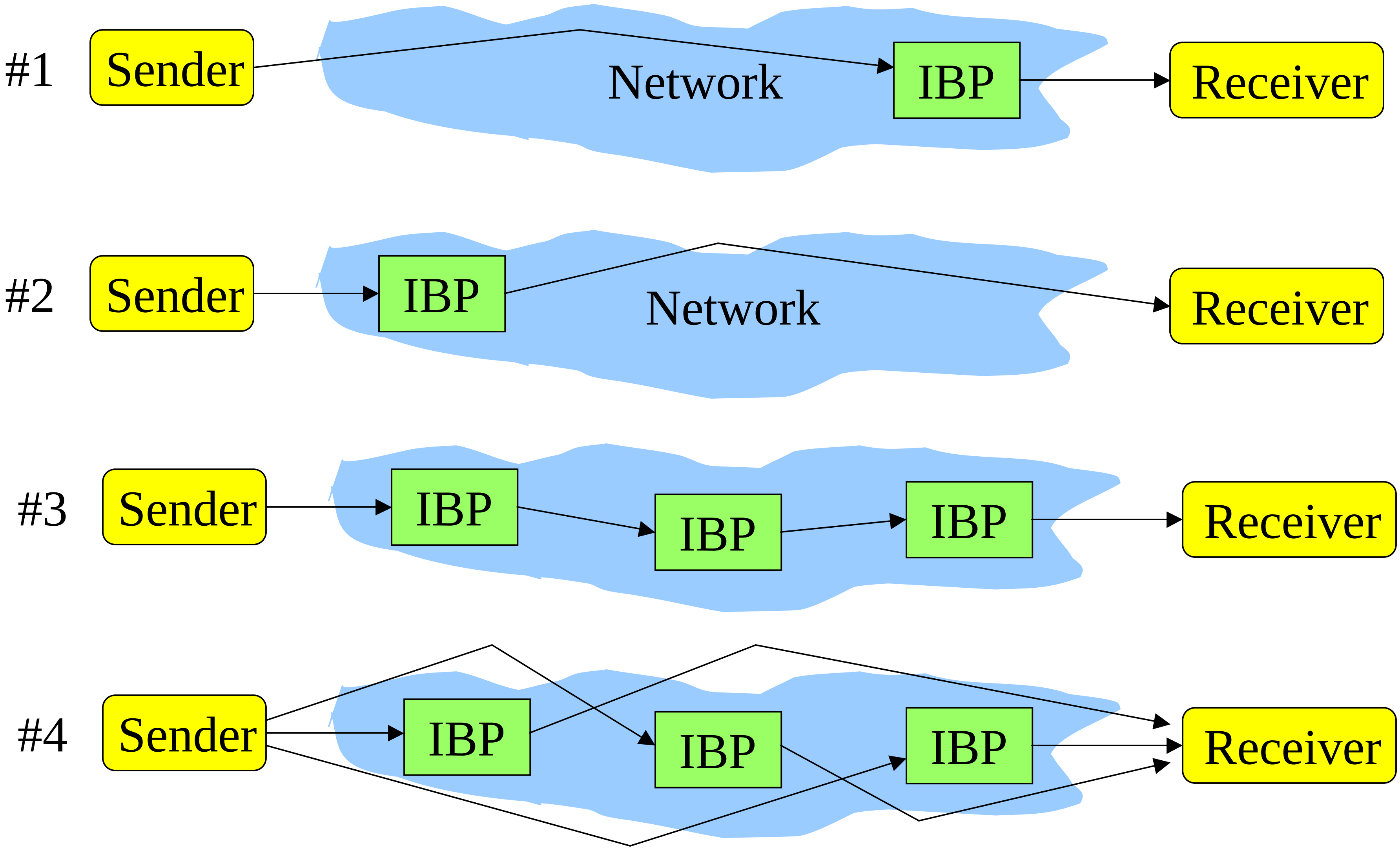
What is IBP?

- Managing and using state in the network.
- Inserting storage in the network so that:
  - Applications may use it advantageously.
  - Storage owners do not lose control of their resources.

Typical IBP usage scenario



Logistical Networking Strategies



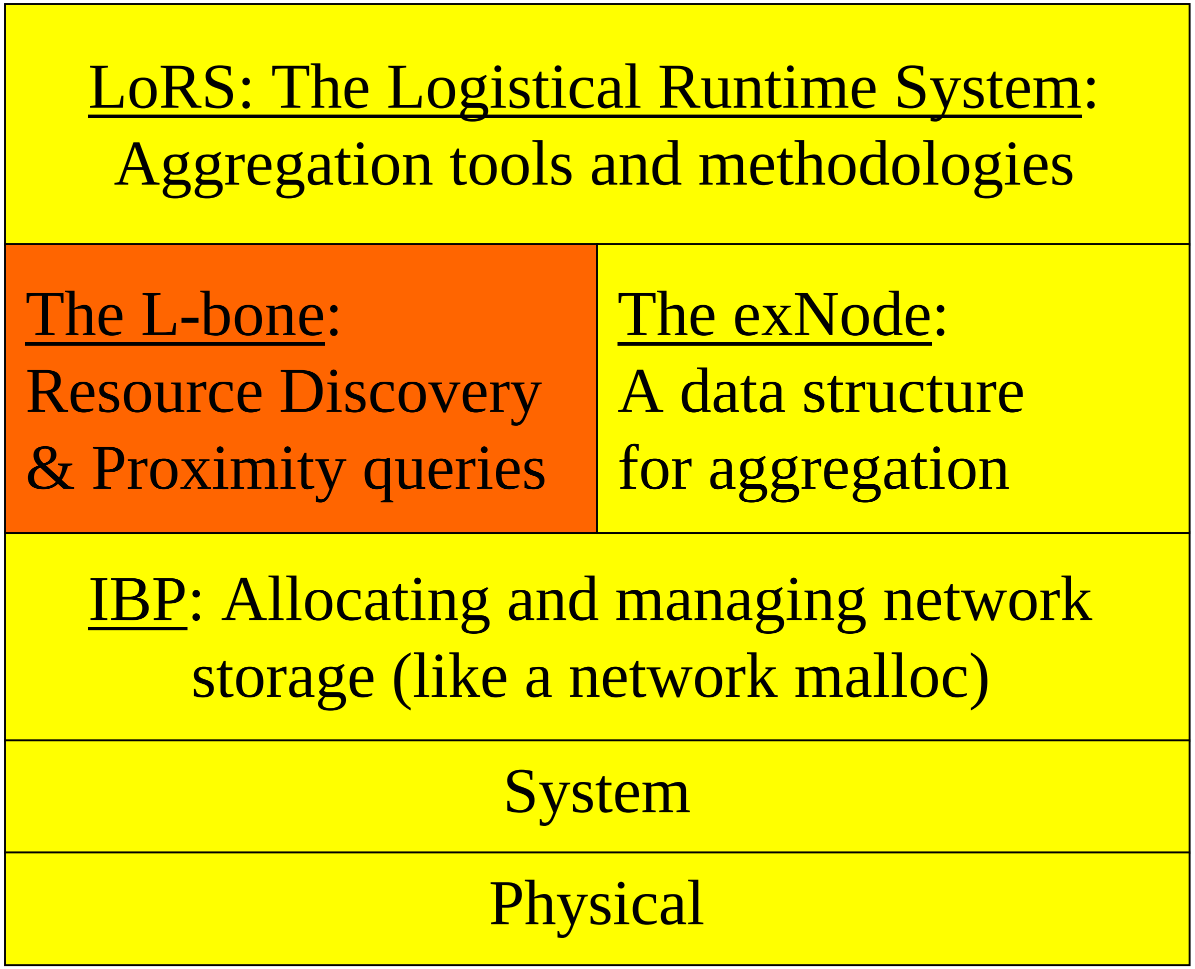
IBP: Slight Detail

Low-level primitives and software for allocating and using *time-limited, append-only* storage buffers in the network:

- Allocate: Like a network malloc()
- Read/Write/Copy (capability-based)
- Manage

Base functionality for logistical networking.

The Network Storage Stack



LoRS: The Logistical Runtime System: Aggregation tools and methodologies

The L-bone: Resource Discovery & Proximity queries

The exNode: A data structure for aggregation

IBP: Allocating and managing network storage (like a network malloc)

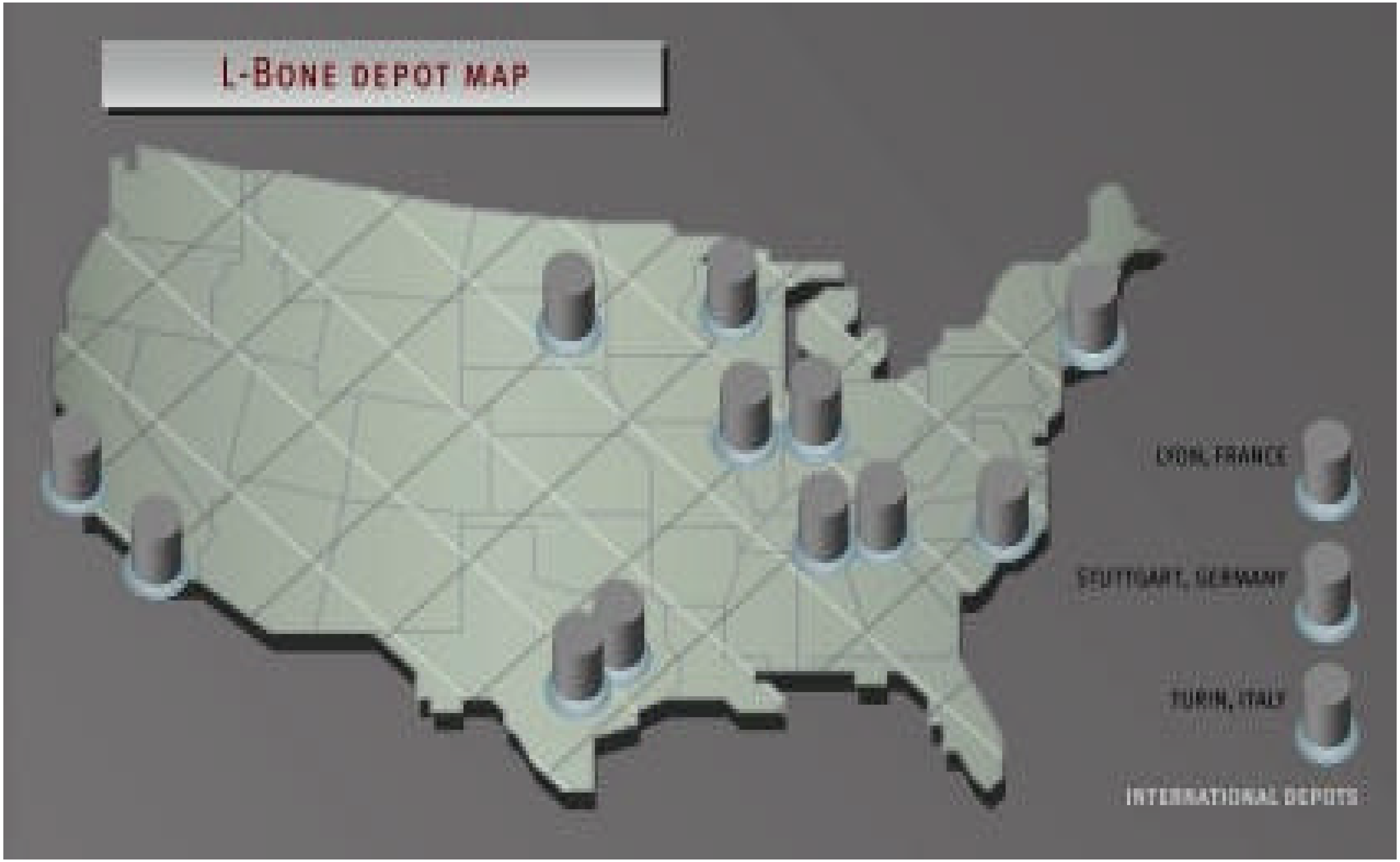
System

Physical

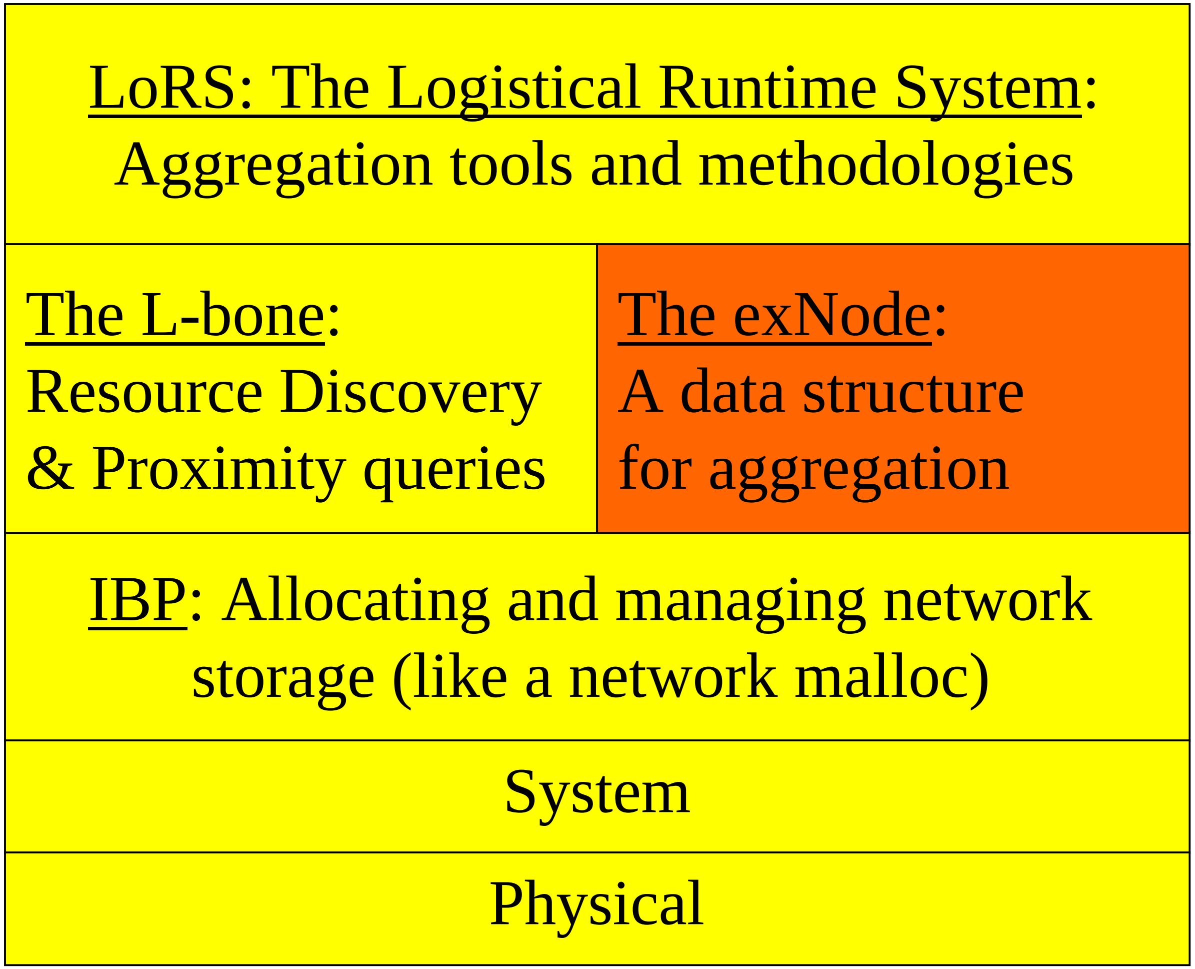
The Logistical Backbone (L-Bone)

- LDAP-based storage resource discovery.
- Query by capacity, network proximity, geographical proximity, stability, etc.
- Periodic monitoring of depots.
- Roughly 1 Terabyte of publicly accessible storage. (scaling to a petabyte someday...)

Snapshot: May, 2002



The Network Storage Stack



LoRS: The Logistical Runtime System: Aggregation tools and methodologies

The L-bone: Resource Discovery & Proximity queries

The exNode: A data structure for aggregation

IBP: Allocating and managing network storage (like a network malloc)

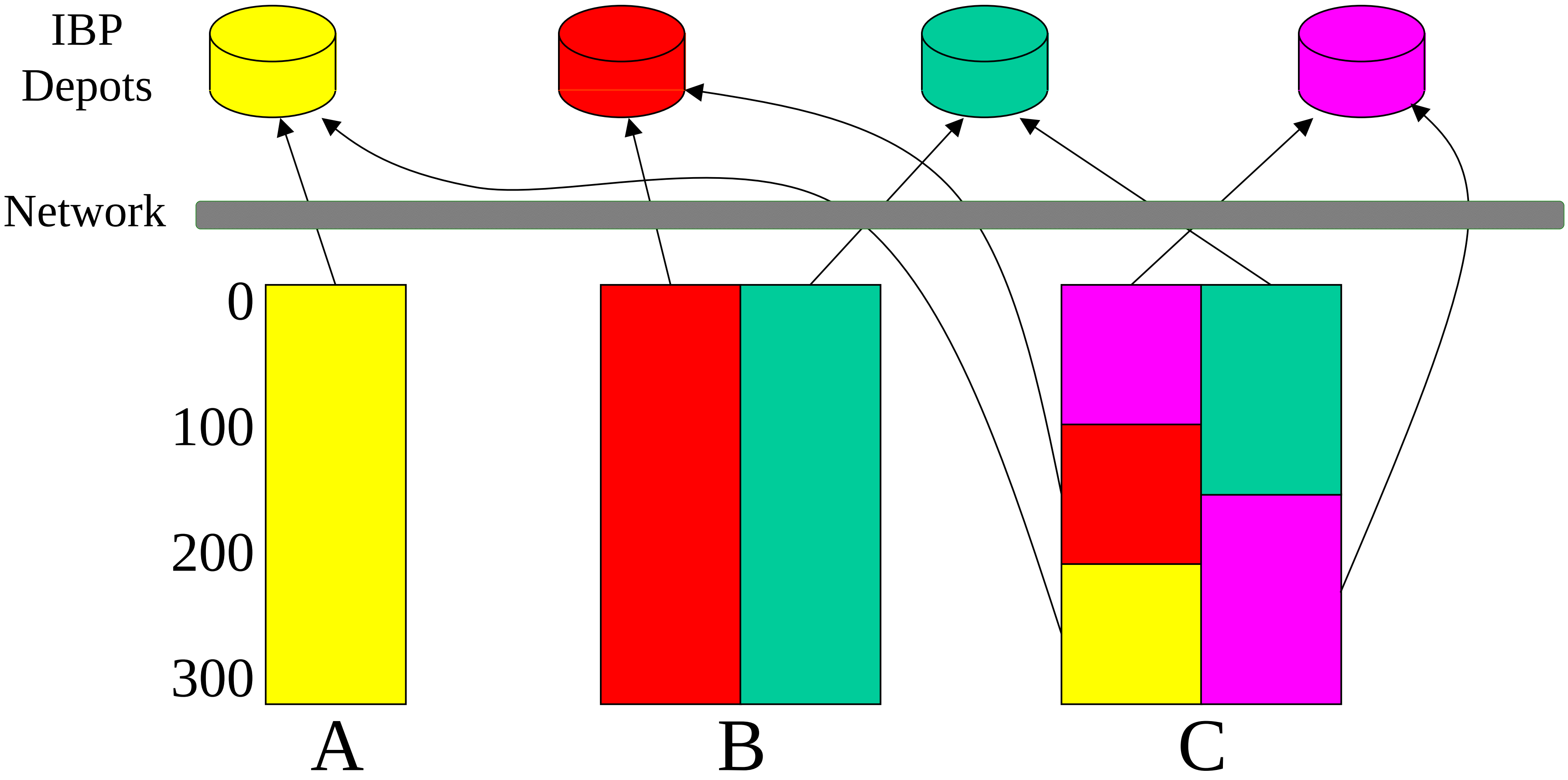
System

Physical

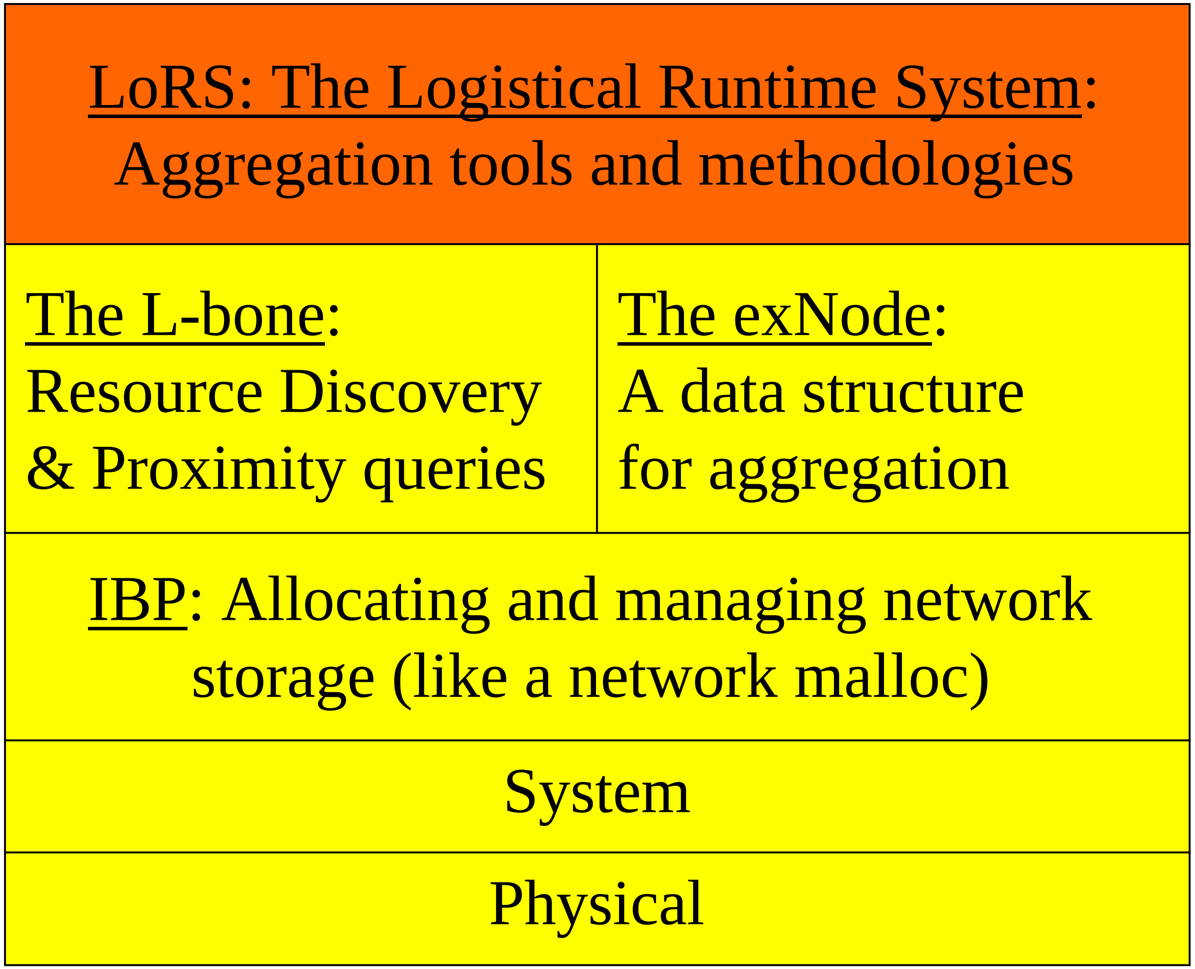
The exNode

- The Network “File” Pointer
- XML-based data structure/serialization
- Map byte-extents to IBP buffers (or other allocations).
- Allows for replication, flexible decomposition of data.
- Also allows for error-correction/checksums
- Arbitrary metadata.

The exNode (XML-based)



The Network Storage Stack



LoRS: The Logistical Runtime System: Aggregation tools and methodologies

The L-bone:
Resource Discovery
& Proximity queries

The exNode:
A data structure
for aggregation

IBP: Allocating and managing network storage (like a network malloc)

System

Physical

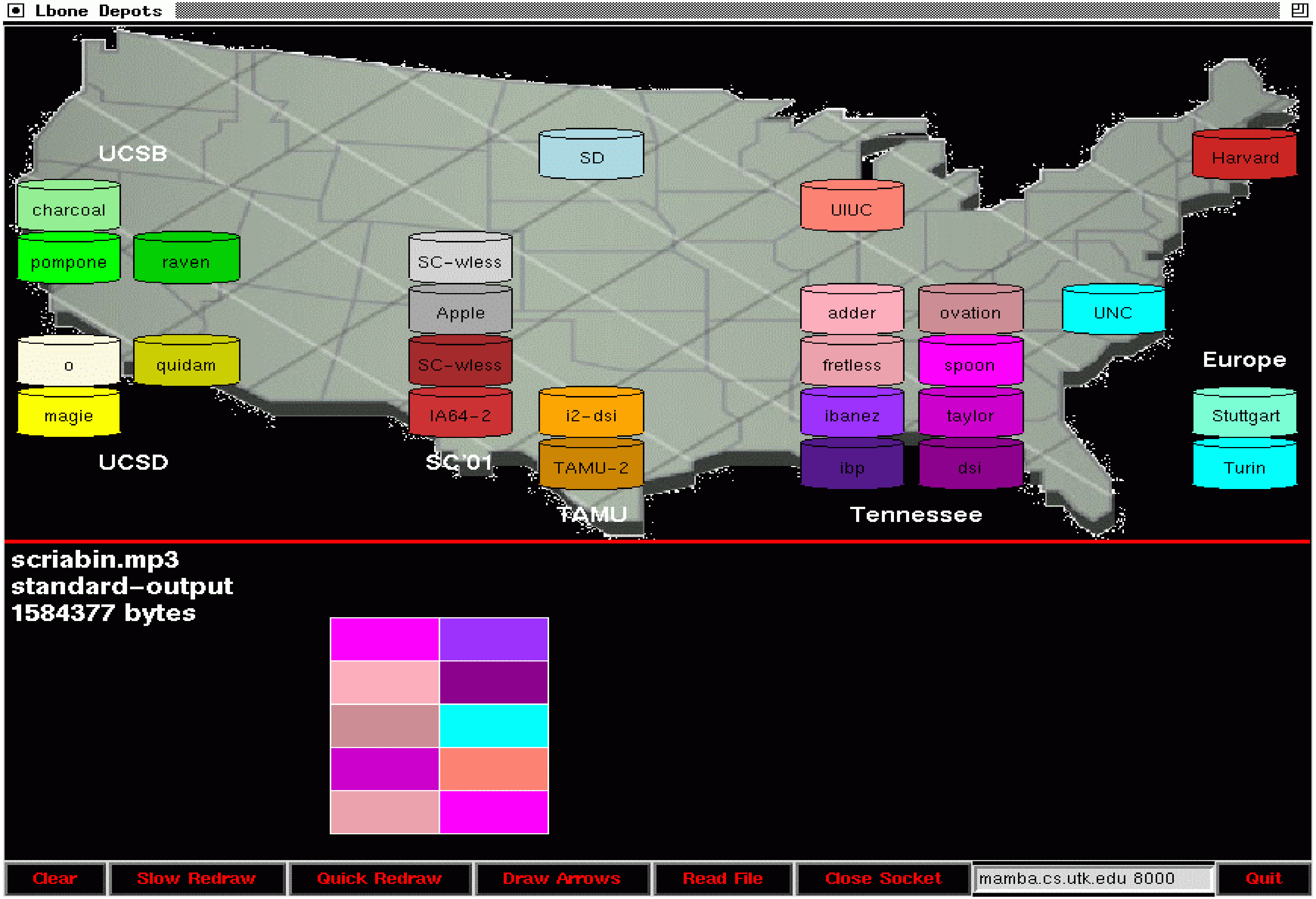
Logistical Runtime System

- Aggregation for:
  - – Capacity
  - – Performance (striping)
  - – More performance (caching)
  - – Reliability (replication)
  - – More reliability (ECC)
  - – Logistical purposes (routing)

Logistical Runtime System

- Basic Primitives:
  - Upload
  - Download
  - Augment
  - Trim
  - Stat
  - Refresh

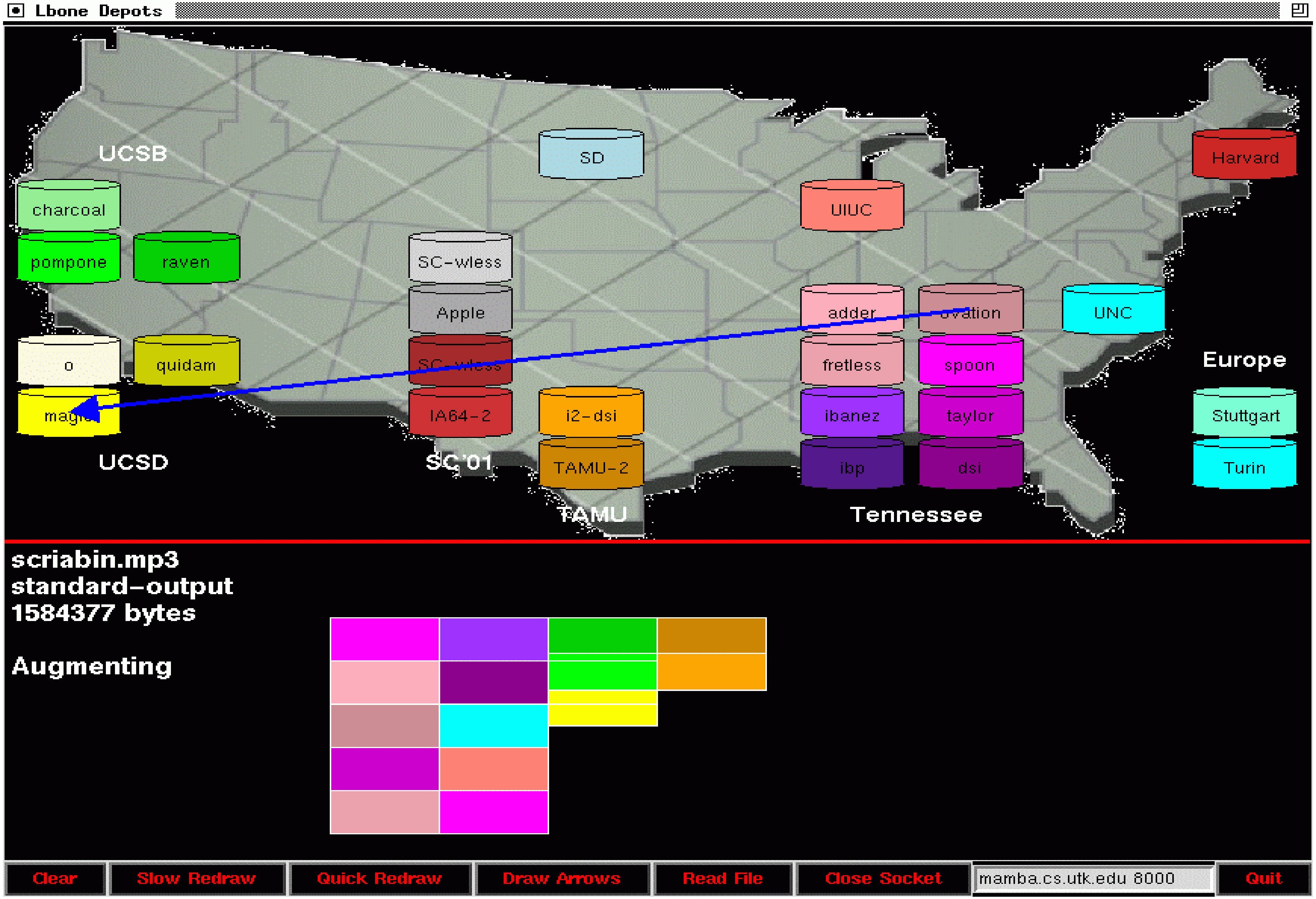
Demonstration: Upload



scriabin.mp3
standard-output
1584377 bytes

mamba.cs.utk.edu 8000

Augment

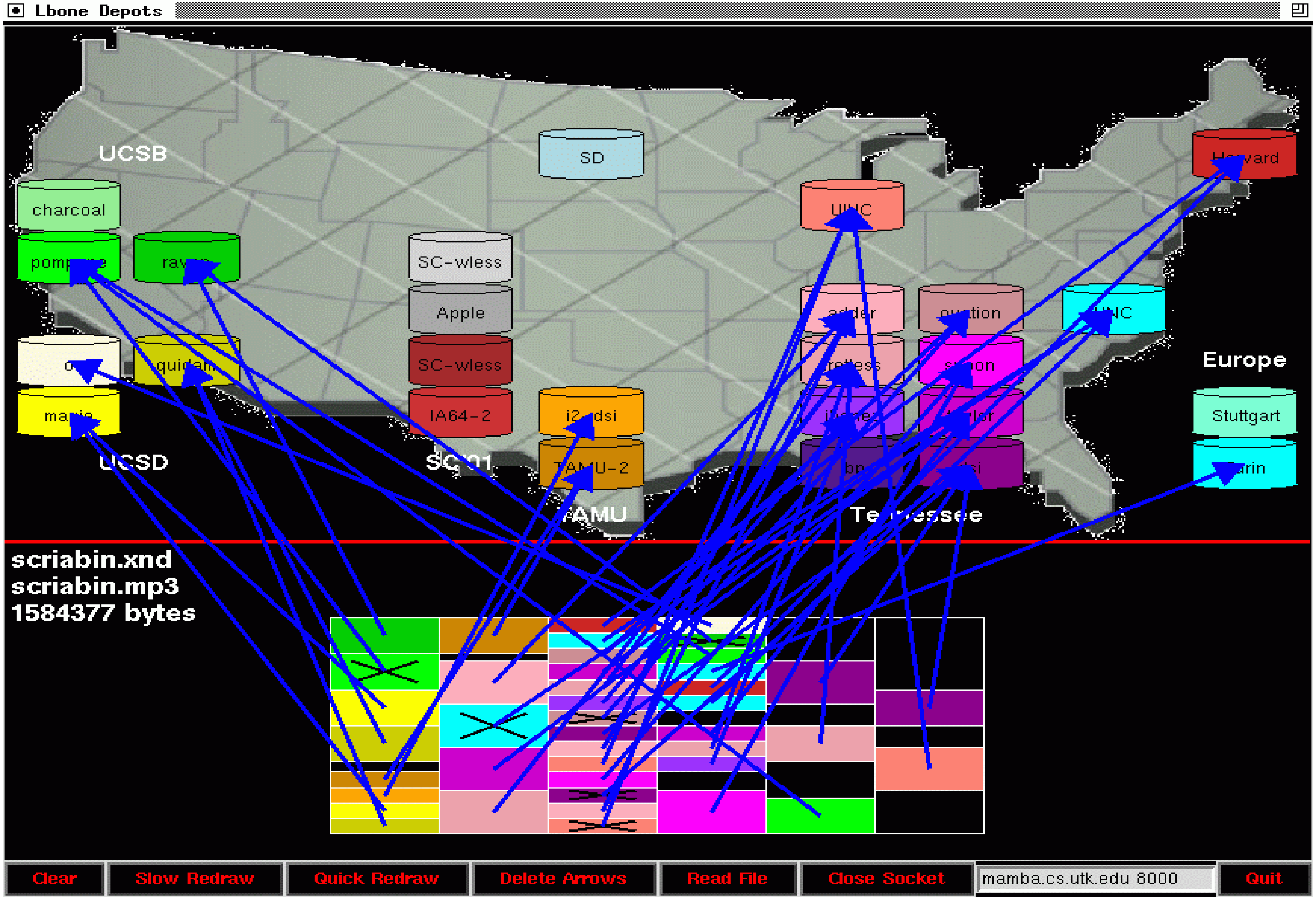


scriabin.mp3
standard-output
1584377 bytes

Augmenting

mamba.cs.utk.edu 8000

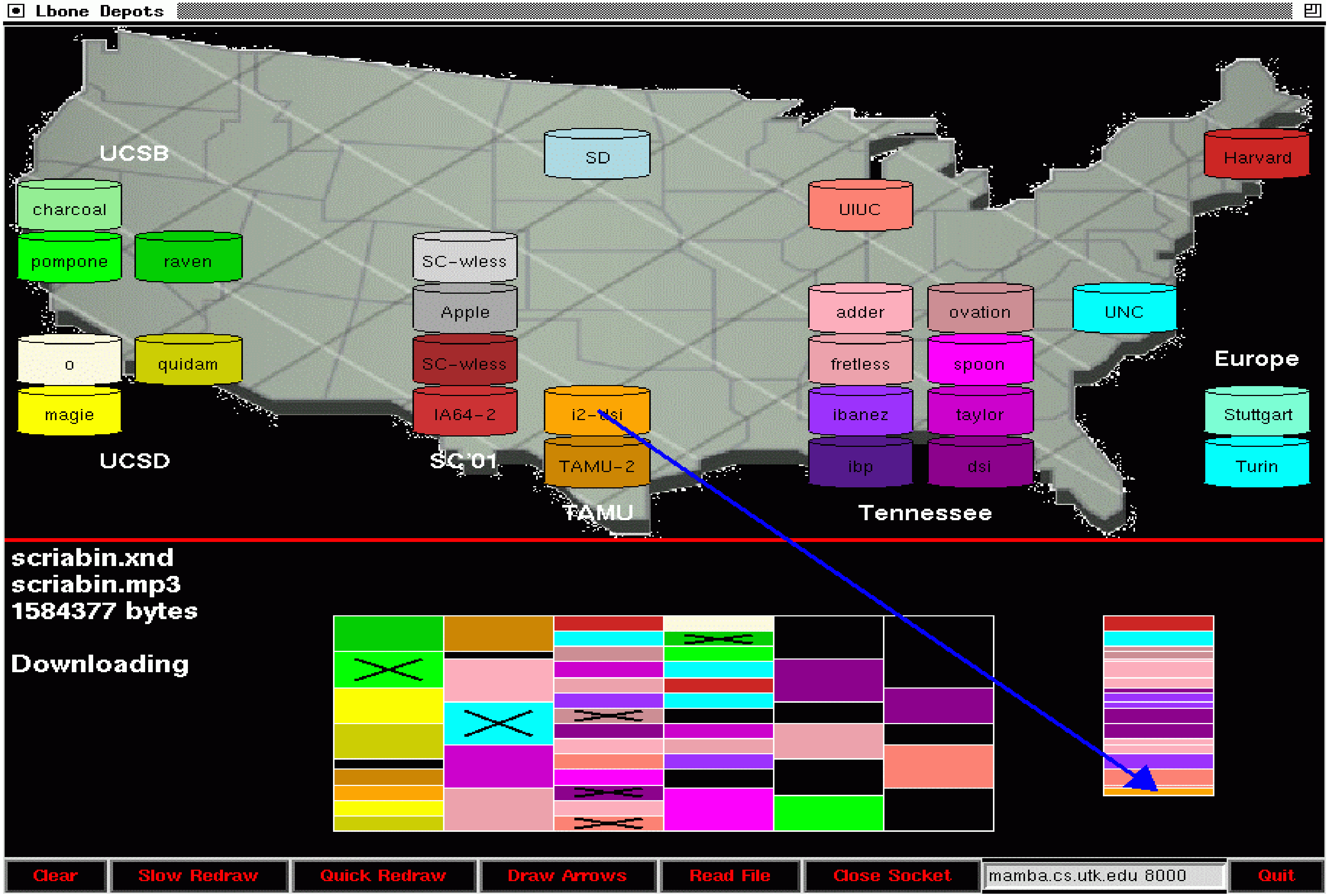
Stat



scriabin.xnd
scriabin.mp3
1584377 bytes

mamba.cs.utk.edu 8000

Download

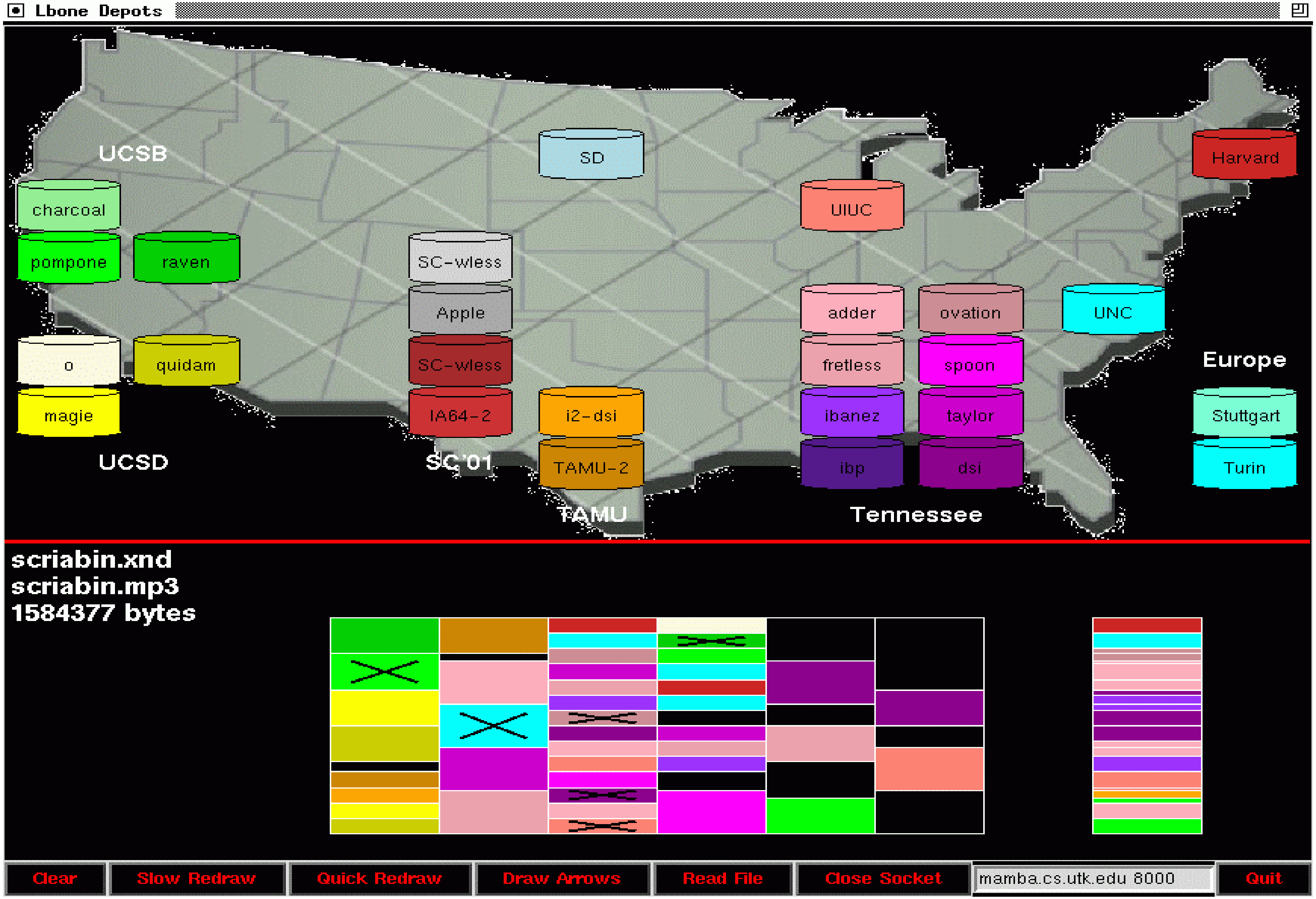


scriabin.xnd
scriabin.mp3
1584377 bytes

Downloading

mamba.cs.utk.edu 8000

Download Finished



scriabin.xnd
scriabin.mp3
1584377 bytes

Where’s The Fault-Tolerance?

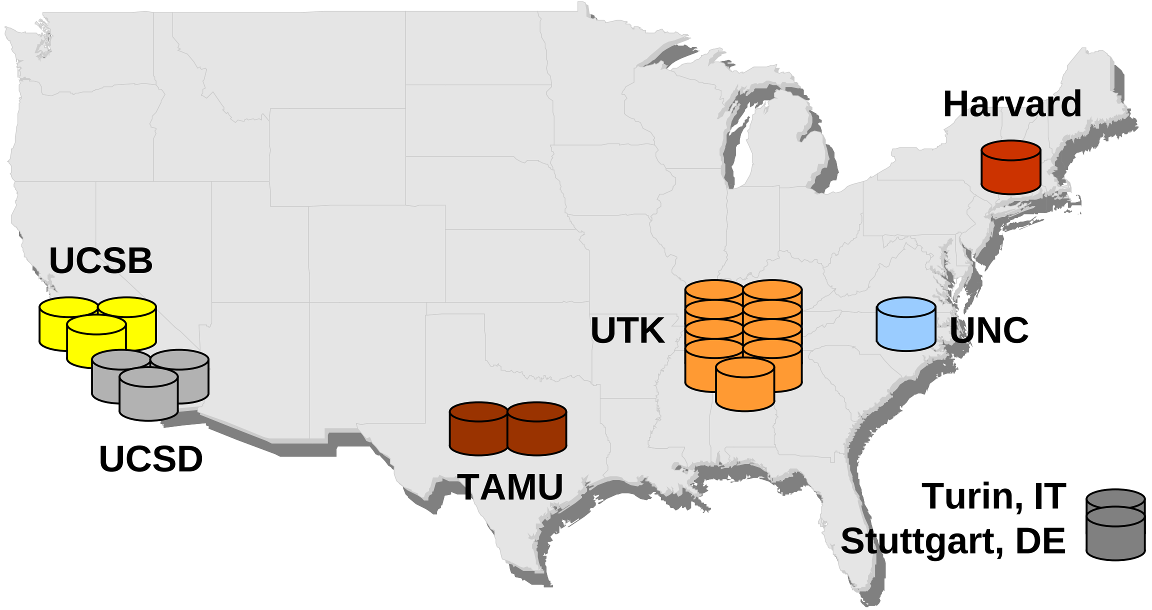
Everywhere

- End-to-end guarantees
- Replication (prediction/monitoring)
- RAID-Like encodings
- Checkpoint support

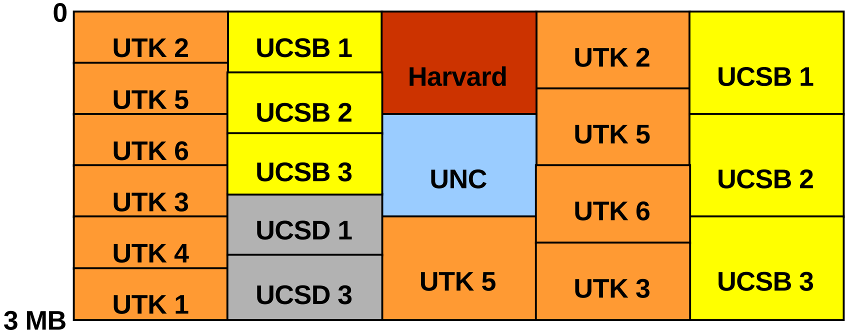
End-To-End Guarantees:

- Checksums stored per exNode block to detect corruption.
- Encryption is an option (DES).
- Compression is an option.

Replication: Experiment #1



3 MB file



UTK 2
UTK 5
UTK 6
UTK 3
UTK 4
UTK 1

UCSB 1
UCSB 2
UCSB 3
UCSD 1
UCSD 3

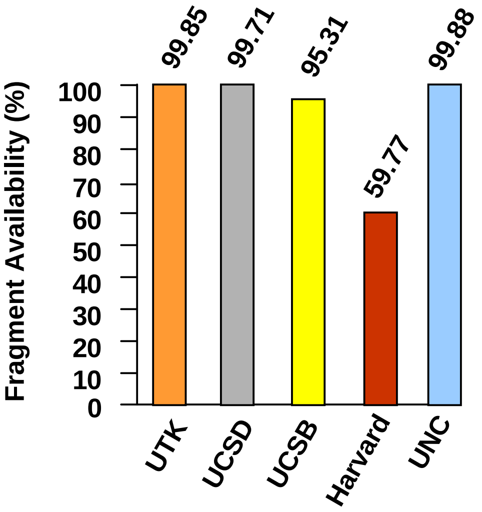
Harvard
UNC
UTK 5

UTK 2
UTK 5
UTK 6
UTK 3

UCSB 1
UCSB 2
UCSB 3

Replication: Experiment #1

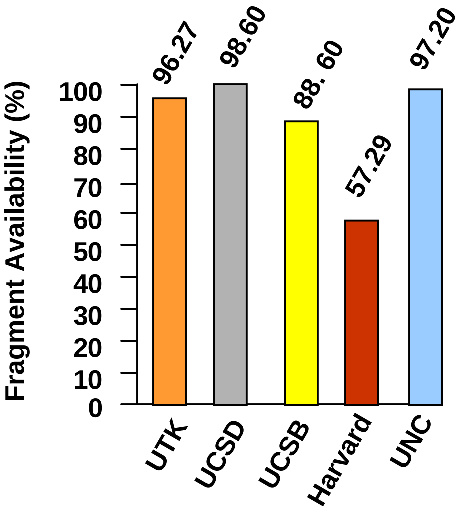
Depot Availability
at UTK



860 Download
Attempts

100% Success

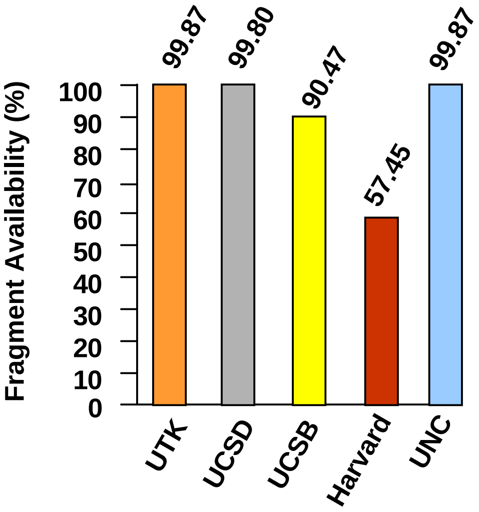
Depot Availability
at UCSD



857 Download
Attempts

100% Success

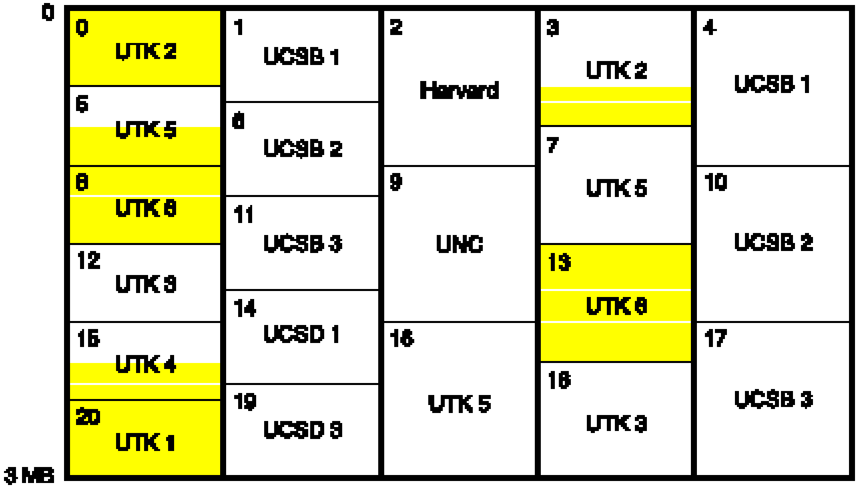
Depot Availability
at Harvard



751 Download
Attempts

100% Success

Most Frequent Download Path


0
3 MB

0 UTK 2
5 UTK 5
8 UTK 6
12 UTK 3
15 UTK 4
20 UTK 1

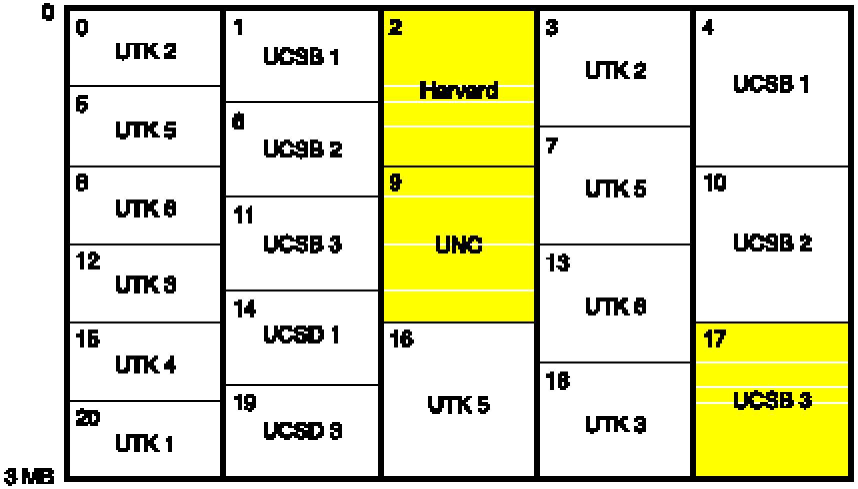
1 UCSB 1
6 UCSB 2
11 UCSB 3
14 UCSD 1
19 UCSD 3

2 Harvard
9 UNC
16 UTK 5

3 UTK 2
7 UTK 5
13 UTK 6
18 UTK 3

4 UCSB 1
10 UCSB 2
17 UCSB 3

From UTK


0
3 MB

0 UTK 2
5 UTK 5
8 UTK 6
12 UTK 3
15 UTK 4
20 UTK 1

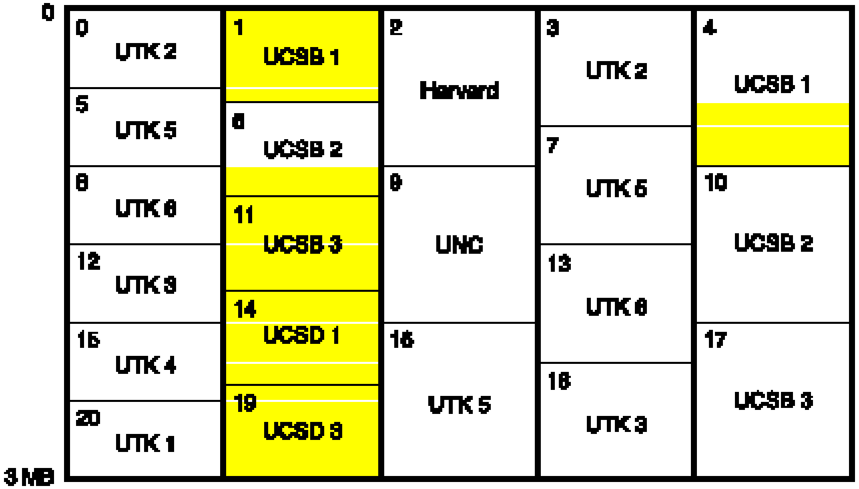
1 UCSB 1
6 UCSB 2
11 UCSB 3
14 UCSD 1
19 UCSD 3

2 Harvard
9 UNC
16 UTK 5

3 UTK 2
7 UTK 5
13 UTK 6
18 UTK 3

4 UCSB 1
10 UCSB 2
17 UCSB 3

From Harvard


0
3 MB

0 UTK 2
5 UTK 5
8 UTK 6
12 UTK 3
15 UTK 4
20 UTK 1

1 UCSB 1
6 UCSB 2
11 UCSB 3
14 UCSD 1
19 UCSD 3

2 Harvard
9 UNC
16 UTK 5

3 UTK 2
7 UTK 5
13 UTK 6
18 UTK 3

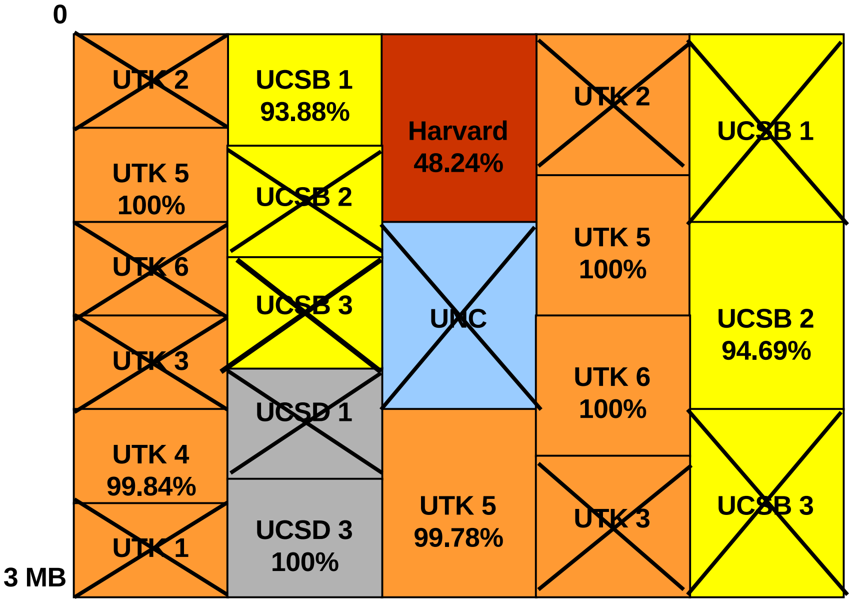
4 UCSB 1
10 UCSB 2
17 UCSB 3

From UCSD

Replication: Experiment #2

- Deleted 12 of the 21 IBP allocations
- Downloaded from UTK

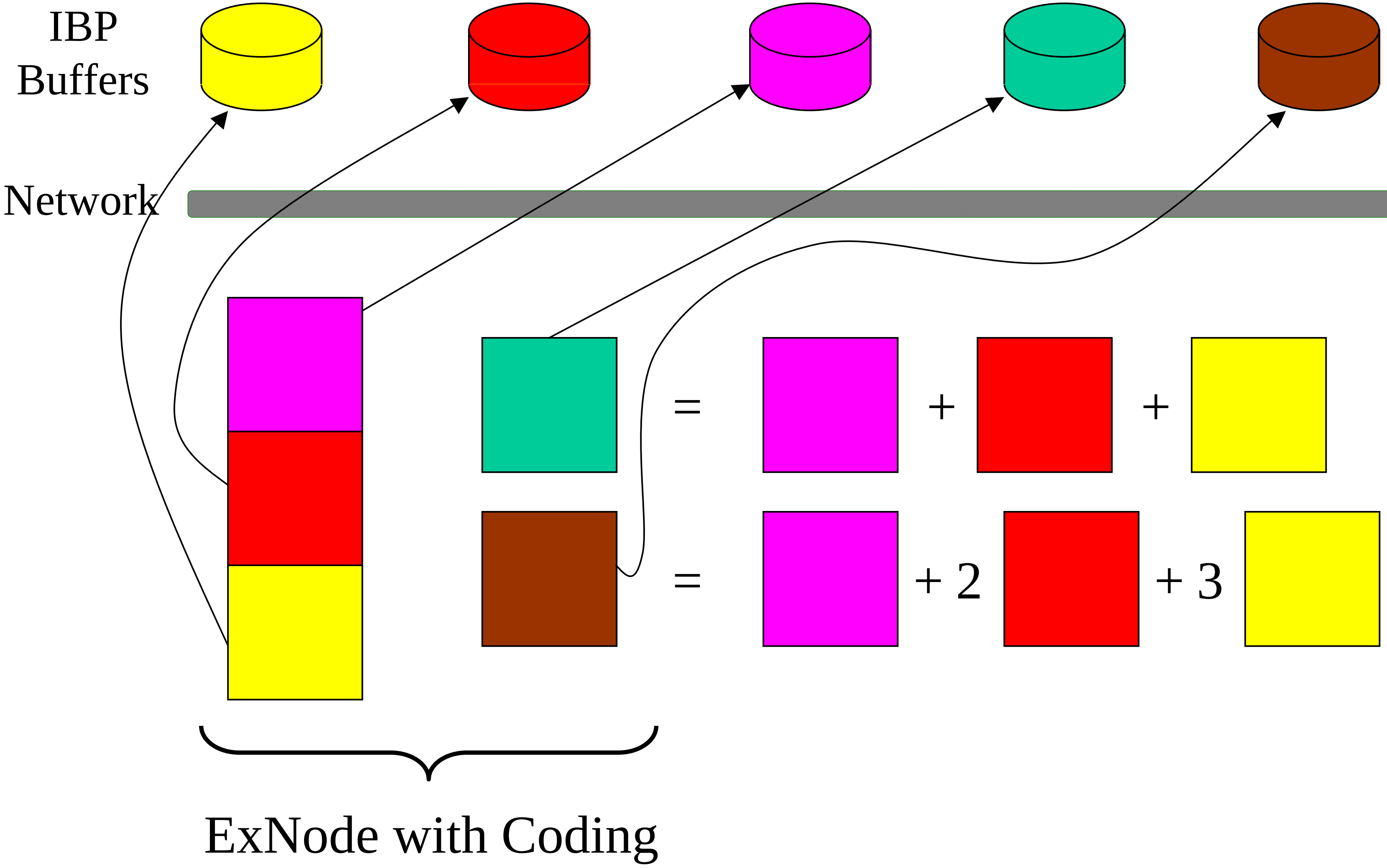
3 MB file



1,225 Attempts

93.88% Success

Coding coming soon



Checkpointing Support

A natural storage substrate for checkpointing

- Time-limited eases garbage collection
- Storage external to computation nodes
- Many-to-many checkpointing operation
- Very flexible

What’s Coming Up?

- More nodes on the L-Bone
- Collaboration with applications groups
- Higher use of lent resources (more faults)
- Logistical File System
- A Computation Stack

- Code / Information at loci.cs.utk.edu

Fault-Tolerance, Network Storage and Logistical Computing

James S. Plank

Director:
Logistical Computing and Internetworking (LoCI) Laboratory

Department of Computer Science
University of Tennessee